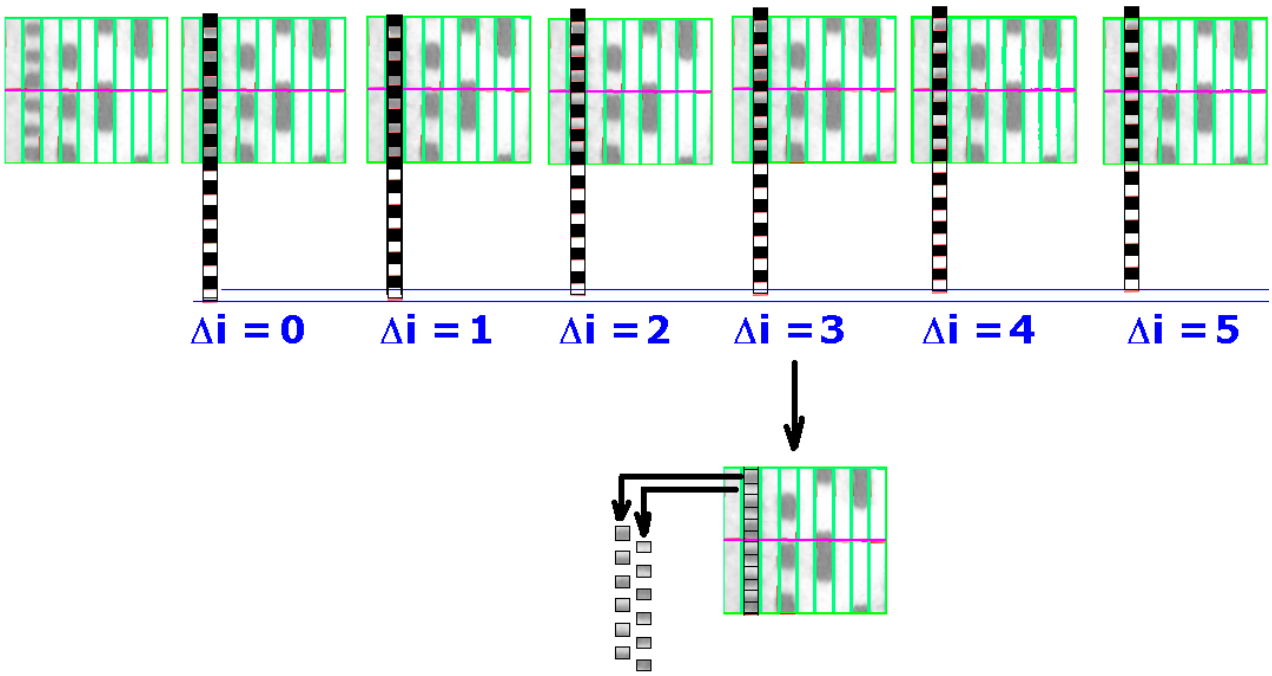


Pour chaque cas on calcule le contraste entre la partie couverte par le niveau haut du modèle et la partie couverte par les niveaux bas (illustration ci-dessus pour le cas $\Delta i=3$). Entre chaque cas, on déplace par pas de 1 pixel le modèle par rapport à l'image.

On note S_h la somme des niveaux des pixels couverte par un niveau haut
S_b la somme des niveaux des pixels couverte par un niveau bas

On obtient la valeur ***C*** du contraste définie par $C = \left| \frac{S_n - S_b}{S_n + S_b} \right|$

Le contraste minimum *(ou maximum, ce qui revient au même à un déphase d'une demi-période)* indique le meilleur alignement entre le modèle et le signal observé.

On réitérè la procédure sur chaque bit de considéré dans cette partie, indépendamment des un des autres.

4.6) Bits de poids forts

Le principe précédent ne peut pas être utilisé pour les bits de poids fort. En effet l'algorithme de minimisation du contraste revient à chercher délibérément la position d'un front de passage d'un niveau haut à un niveau bas (ou inversement). Or, pour les bits de poids fort, il existe des zones ou le bit reste à zéro ou à un sur la totalité du champs de vue de la webcam.

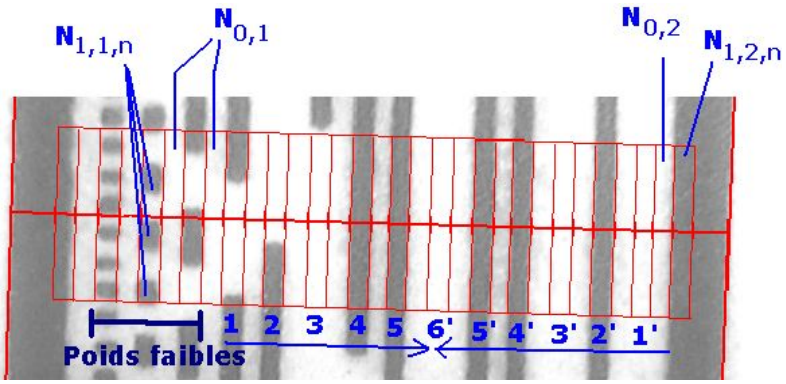
Il faut donc utiliser un algorithme qui détecte un front si il existe, ou détecte un niveau si il n'y a pas de front.
(Etape 0) Il faut dans un premier temps initialiser la valeur des niveaux. On utilise les lignes guides (noirs) et les interlignes (blanches) pour déterminer la valeur des niveaux hauts et bas sur l'image. On considère de plus la valeur des niveaux haut et bas des bits de poids faible.

Pour le coté bits de poids fort Le niveau noir est initialisé à l'aide de la couleur moyenne de la ligne guide de droite. On note la valeur de ce niveau $N_{1,2,n}$. Le niveau blanc correspondant est initialisé sur la couleur moyenne de l'interligne précédent cette ligne guide. La valeur de ce ce niveau est noté $N_{0,2.}$

Pour le coté bits de poids faible. On utilise soit le dernier bit de poids faible ($n_f=1$), soit l'avant dernier bit de poids faible *($n_f=2$). (Le dernier bit de poids faible est utilisé si l'on est certain d'avoir un niveau noir suffisamment grand pour obtenir une bonne estimation de la valeur de ce niveau. Dans le cas contraire on utilise l'avant dernier.)*

La couleur noire moyenne du bit de poids faible retenue est notée $N_{1,1,n}$

Le niveau blanc correspondant est initialisé sur la couleur moyenne des deux interlignes encadrant la ligne du dernier bit de poids faible. La valeur de ce ce niveau est noté $N_{0,1}$

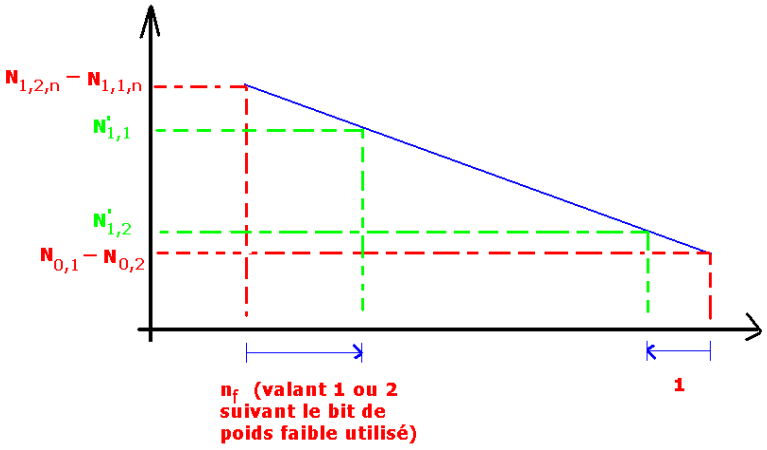


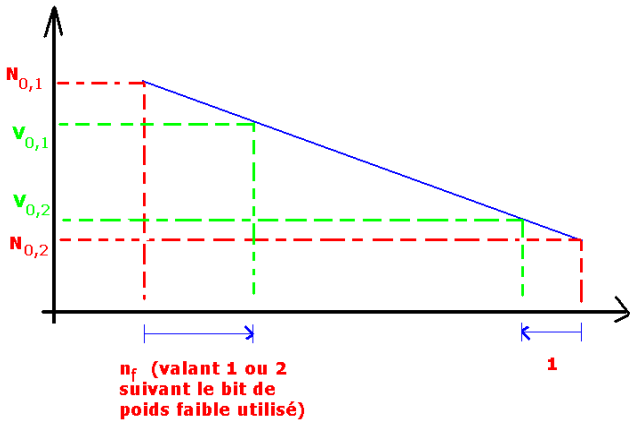
Finalement, afin de tenir compte du gradient de luminosité on interpole la valeur des niveaux noirs et blancs dans la zone du bit considérée.

Les valeurs retenues pour les niveaux noirs sont:

$$V_{1,1} = 0,7\ N'_{1,1} + 0,3\ N'_{0,1}$$
$$V_{1,2} = 0,7\ N'_{1,2} + 0,3\ N'_{0,2}$$

Avec $N'_{1,1}$ et $N'_{0,1}$ les notations introduites dans la figure ci-contre.

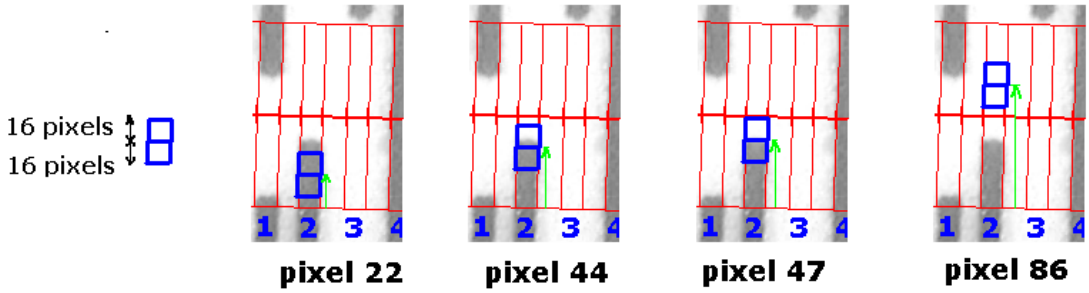




Les valeurs retenues pour les niveaux blancs ($V_{0,1}$ et $V_{0,12}$) sont interpolés comme représentées par la figure ci-contre.

(Etape 1) Le niveau de chaque bit est déterminé deux à deux en partant de l'extérieur vers l'intérieur, comme présenté sur la figure en haut de cette page.
On recherche un front en moyennant les valeurs sur 16 pixels en amont et en aval du front supposé. On obtient ainsi deux couleurs qui, si elles sont suffisamment différentes, marquent la présence d'un front. La différence de couleur droit d'au-moins 90% de différence de $V_{1,i}$ et $V_{0,i}$, i valant 1 pour le côté poids faible, 2 pour le côté poids fort. La plus grande différence de valeur est la position effective du front.

Si on ne trouve pas différence de couleur suffisante, alors il n'y a pas de front et le niveau du bit et pris bas ou au en fonction de la couleur moyenne de ce niveau relativement à $V_{1,i}$ et $V_{0,i}$.



pixel 22 **pixel 44** **pixel 47** **pixel 86**

L'image ci-dessus, illustre le principe. Dans cet exemple, le pixel 22 (ou 86) ne présente pas de différence de couleur important dans les deux zones. Au contraire, le pixel 44 présente une différence mais comme le pixel 47 présente une différence encore plus importante, le front sera localisé au niveau du pixel 47.

(Etape 2) Finalement, on redétermine la valeur correspondant au noir et blanc. On utilise les valeurs des interlignes blanches encadrant les bits considérés à l'étape précédente. Pour la valeur du noir, on considère le niveau noir des bits précédents si il a été détecté sinon on conserve la valeur précédente. Pour les deux couleurs, noirs et blanche, on tient compte d'un éventuelle gradient de luminosité en utilisant la même méthode que dans l'étape d'initialisation des valeurs (Etape 0).

4.7) Moindres carrés sur un ensemble de bits

4.7.1) Calcul préliminaire

Les paragraphes §4.5 et $4.6 ont permis de localiser la position des différents bits du code Gray. Cela revient à déterminer les valeur des $x'_{n,b}$ du paragraphe §2.4., on prenant un référence quelconque.

Si l'on considère deux bits successifs, on a: $x'_{n,j-1} = (4n+1)\ e\ 2^{j-1}$

et $x'_{n,j} = (4n+1)\ e\ 2^{j}$

Remarque : l'orientation de l'axe des ordonnées sur le fichier d'impression du code gray et opposé à celui de l'axe des ordonnées sur l'image d'une webcam. Ainsi pour la suite on prendra la notation correspondant à l'orientation de la webcam soit:

$x'_{n,j-1} = -(4n+1)\ e\ 2^{j-1}$

et $x'_{n,j} = -(4n+1)\ e\ 2^{j}$

et, en particulier, pour n=0 pour le bit j et n=k pour le bit j-1, on obtient:

$x'_{k,j-1} = -(4k+1)\ 2^{j-1}$

et $x'_{0,j} = -e\ 2^{j}$

d'où $x'_{0,j-1} - x'_{k,j} = e\ 2^{j} - (4k+1)\ e\ 2^{j-1}$

et ainsi $k = \frac{x'_{0,j-1} - x'_{k,j}}{2^{j+1}\ e} + \frac{1}{4}$ *(on rappelle que k est un entier)*

4.7.2) Déterminant de la position du code en haute précision

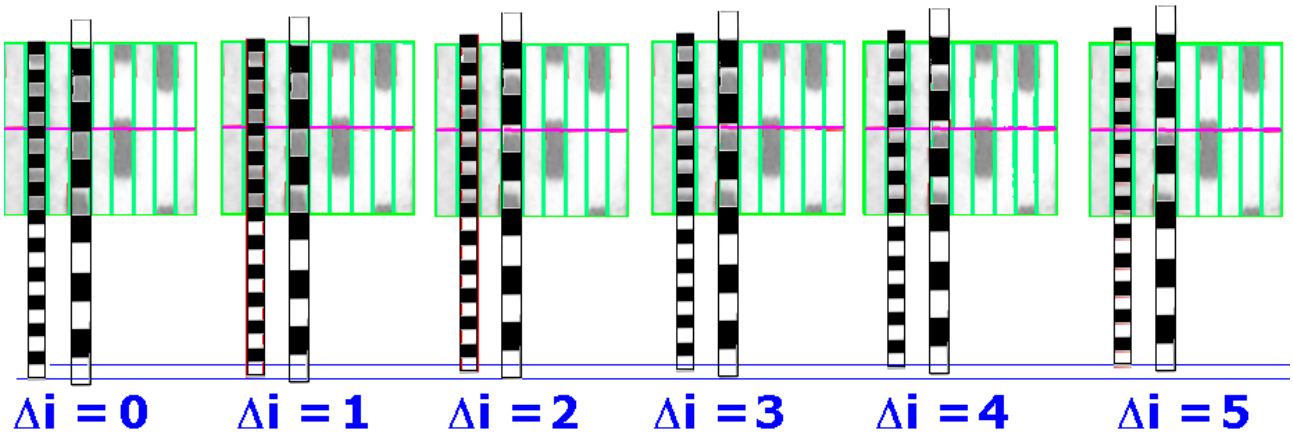
Bits de poids faible

D'après le calcul du paragraphe précédent, les positions des bits de poids faible sont liées par une relation faisant apparaître une valeur entière noté k.

Les valeurs correspondent aux ***x'*** déterminés par la méthode du §4.5 ont été trouvées indépendamment des une des autres.

Dans un premier temps on recalcule par récurrence la valeur de ***x'*** en utilisant les valeurs trouvées au §4.5 tout en imposant une valeur de k entière la plus proche de celle trouvée. Seule le bit de poids le plus faible conserve la valeur de ***x'*** trouvée par la méthode §4.5. Dans la pratique les autres ***x'*** changent peut, la méthode §4.5 étant suffisamment fiable.

On utilise ensuite la méthode décrite au paragraphe §4.5, cette fois-ci on minimisant le contraste sur l'ensemble de bits de poids faible en un seul bloc. De plus, le modèle se déplace par pas de 0,1 pixel par rapport à l'image.



La détermination de la position du code avec une précision de 0,1 pixel est équivalant au principe du vernier sur un pieds à coulisse. En effet, la zone observée couvre 128 pixels. L'ordre de grandeur de l'échelle étant de 5 pixels, on couvre de l'ordre d'une dizaine d'une période du bits de poids faible. On peut donc mesurer la position du code en dessous d'une échelle de 1 pixel.

Bits de poids fort

On reprend le calcul préliminaire, mais en considérant cette fois-ci des bits non successif noté i et i-n.

$$x'_{k,j-n} = -(4k+1)\ 2^{j-n}$$

$$\text{et} \quad x'_{0,j} = -e\ 2^{j}$$

d'où $x'_{0,j-n} - x'_{k,j} = e\ 2^{j} - (4k+1)\ e\ 2^{j-n}$

et ainsi $k = \frac{x'_{0,j-1} - x'_{k,j}}{2^{j-n+2}\,e} + 2^{n-2} - \frac{1}{4}$ *(on rappelle que k est un entier)*

On redétermine la valeur de k des bits de poids fort en utilisant la relation ci-dessus et en considérant la valeur des ***x'*** des bits de poids comme référence. Par défaut, la valeur de n est 1, c'est a dire que l'on recalcule le x' d'un bit en utilisant la valeur de x' du bit qui le précédè. Toutefois, si une valeur de k est trop grande, compte tenu de la grande période des bits de poids fort, on risque d'introduire des erreurs. On incrémente n autant de fois que nécessaire afin d'avoir une valeur pour k suffisamment petite.

4.8) Détermination « haute précision » de la valeur lue

Les paragraphes précédents ont permis de déterminer les valeurs du code Gray sur les deux lignes de lecture.

Lire à la précision du code Gray signifie que nous arrivons à déterminer la position de ce code par rapport à la webcam avec un précision valant l'échelle e introduite au §4.5. Toutefois, nous avons vu que nous étions capable de déterminer la position de code à une précision plus grande (cf. $4.7.2). Nous pouvons donc interpoler la valeur lue entre deux valeurs entières de graduation codée par le code gray.

Ainsi, la valeur entière de la graduation sera déterminer par la conversion du code gray en valeur décimale. Le code gray considéré étant un code 14 bits, cette valeur sera comprise entre 0 et 2^{14} - 1 = 16383.

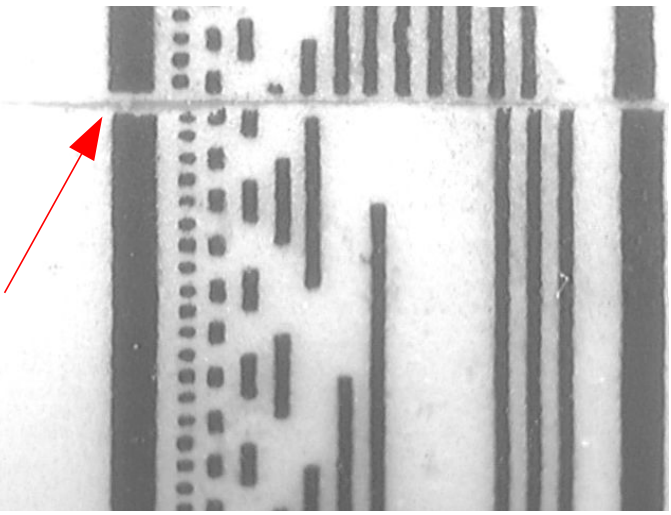
La valeur décimale est réalisée grâce à une interpolation sur la valeur de $x'_{0,0}$ du bit de poids faible.

La valeur décimale lue V_r est donc $\boxed{V_r = V + f}$ ou V est la valeur entière et f la partie décimale de la graduation $(0 \le f < 1)$

4.9) Détrompeur

La méthode présentée comporte un défaut dans le cas ou l'interprétation de l'image de la webcam conduit à une valeur erronée sur l'une des deux lignes de lecture.

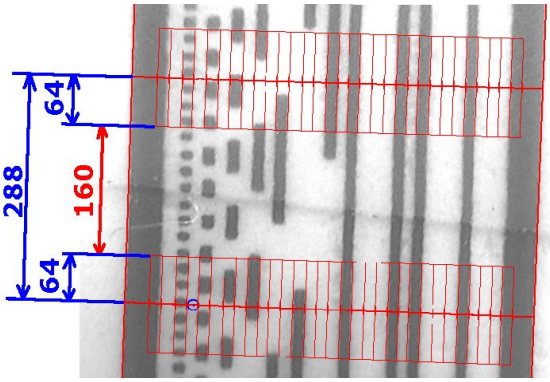
Si l'on considère uniquement deux lignes de lecture et que l'un d'entre elle donne une valeur fausse, l'autre aura peut-être une valeur correcte mais nous serons incapable de dire de façon systématique qui des deux donnent une valeur raisonnable.



Ce cas doit être considéré car il existe nécessairement au moins une discontinuité dans les graduations. Cette discontinuité est constitué par la jonction des deux extrémités de la bande imprimée.

Pour s'affranchir de la discontinuité, il faut que celle-ci ait une longueur maximale $\boldsymbol{G_{max}}$ plus faible que l'espacement entre les deux zones de lecture centrées sur les lignes de lecture soit:

$$G_{max} = 288 - 64 - 64$$
$$G_{max} = 160$$

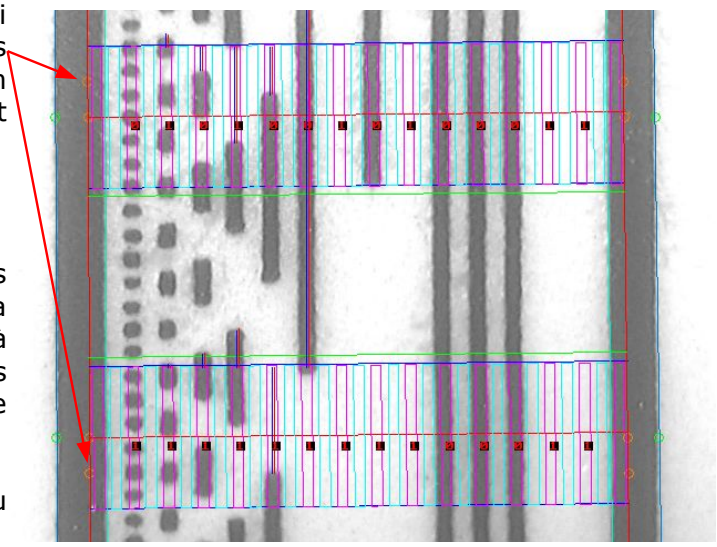


Toutefois pour éviter des effets de bords, la valeur effective $\mathbf{G_{eff}}$, sera prise à $\boldsymbol{F_d}$, F_d=90% de cette dernière soit:

$$G_{eff} = 0{,}9 \times 160$$
$$G_{eff} = 144$$

Cette zone est repérée par les deux lignes vertes de l'image suivante.

Afin de réaliser le détrompeur, on défini deux lignes de lecture secondaires localisées suivants lignes de la webcam 32 pixels avant la ligne de lecture 96 et 32 pixels après la ligne de lecture 384.



La lecture de ces lignes secondaires repose sur le même principe que la lecture des lignes de lecture initiales à ceci-près qu'au lieu de centrée des zones de lecture de 128 pixels, on considère des zones de 64 pixels.

Elles correspondent, pour les notations du $4.3 à:

k=2 pour la ligne guide de gauche et ligne de lecture secondaire du haut
k=6 pour la ligne guide de droite et ligne de lecture secondaire du haut
k=3 pour la ligne guide de gauche et ligne de lecture secondaire du bas
k=7 pour la ligne guide de droite et ligne de lecture secondaire du bas

Ainsi, ce choix permet de garantir qu'elle est la ligne de lecture qui donne une valeur correcte. La ligne de lecture donnant la bonne indication est celle dont la ligne de lecture secondaire donne la même valeur (modulo le décalage de plus ou moins 32 pixels).

V) Étalonnage du code gray

5.1) Présentation

Il ne faut pas se faire d'illusion, imprimer un code Gray sur une imprimante ne permettra pas d'avoir des graduations régulières. La mise en place de la bande imprimé apportera des contrainte de régularité, de tension. Les distorsions optiques de la webcam ne seront pas parfaitement corrigées.

Pour un ensemble de raison, il est nécessaire d'étalonner le code du disque afin de compenser les irrégularités des graduations.

5.2) Principe

Afin de réaliser l'étalonnage nous devons utiliser une longueur référence. Le principe de lecture du code par la webcam nous donne une telle référence.

Nous réalisons deux lectures sur le même champ de vue (voir §2.3). La distance entre les deux lignes de lecture est fixée à 288 pixels. Ainsi, quelle que soit la position du disque de lecture cette distance est de 288 pixels.

La référence fixe choisie est le nombre de pixels nécessaires afin de réaliser un tour complet du disque d'encodage.

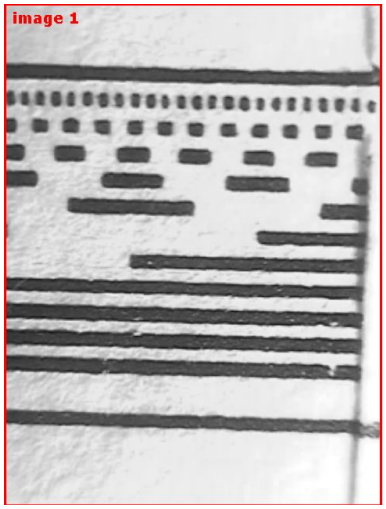
Ainsi un tour complet est de l'ordre de 80785 pixels (voir §1)

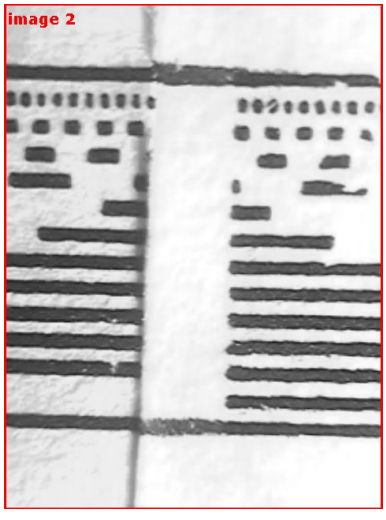
Nous numérotons les graduations du code Gray en leur attribuant un numéro de pixel depuis une origine choisie à priori.

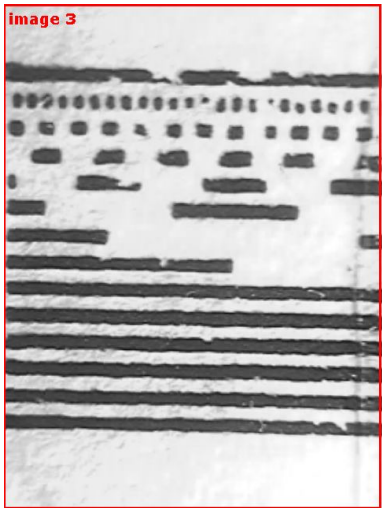
Cela nécessite de prendre des images du code gray, puis de les « assembler » afin de comptabiliser le nombre exacte de pixels nécessaires pour faire un tour complet. La correction des distorsions optiques de la webcams intègrent aussi la courbure du disque. Ceci a pour conséquence que la webcam voit le code gray de façon totalement plane. L'assemblage des images du code gray sera donc une bande plane comme si nous avions déroulé la bande portant le code gray.

Exemple sur 3 images de 640x480 avec une discontinuité dans les graduations:

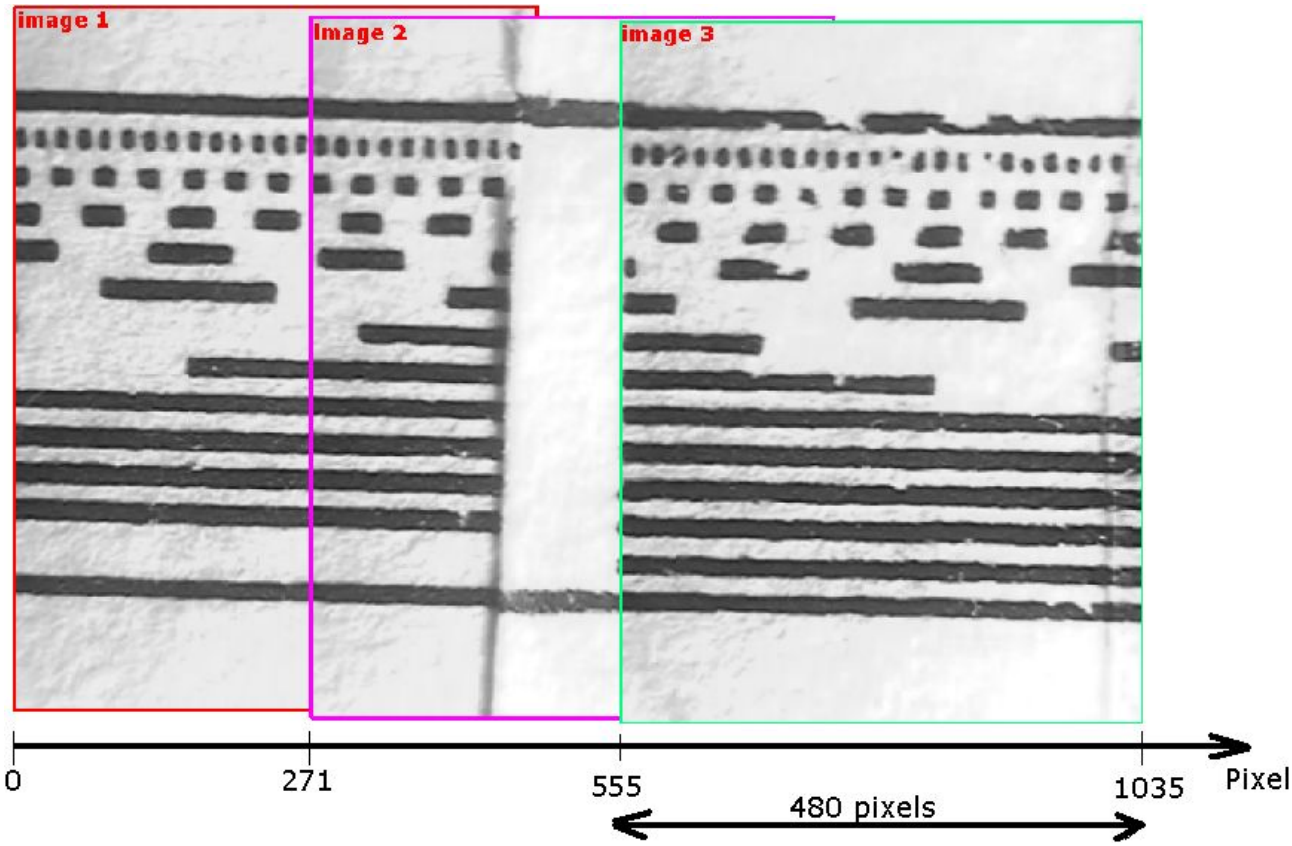
Vues individuelles







Vues assemblées avec numérotation des pixels de l'ensemble:



5.3) Erreur d'assemblage et conséquence sur la précision de l'étalonnage

L'axe du code gray est suivant la largeur de la webcam. Ainsi, chaque prise de vue du code gray comporte 480 pixels.

Or 80785 / 480 = 169 arrondi à l'unité par excès

Il faut donc au minimum de l'ordre de 169 images pour couvrir l'ensemble du code.

Supposons qu'il y ait un décalage de quelques pixels sur l'alignement de chaque image. Ceci conduit à une incertitude sur le nombre de pixels nécessaires pour faire un tour complet.

Toutefois quel que soit ce nombre de pixels un fois le tour du code effectué, nous retrouvons notre point de départ.

Ainsi, si on note **n** le nombre réel de pixels pour faire un tour (par essence ce nombre n'est pas connu), **n'** le nombre mesuré par le programme, **θ** la résolution théorique, et **θ'** la résolution effective de l'étalonnage.

$$n\,\theta = 360° \quad \text{et} \quad n'\,\theta' = 360°$$

Si on note $\Delta\theta = \theta' - \theta$ et $\Delta n = n' - n$

nous obtenons $n\,\theta = (n + \Delta n)(\theta + \Delta\theta)$

Ainsi, $\Delta\theta / \theta = (1 + \Delta n / n)^{-1} - 1$

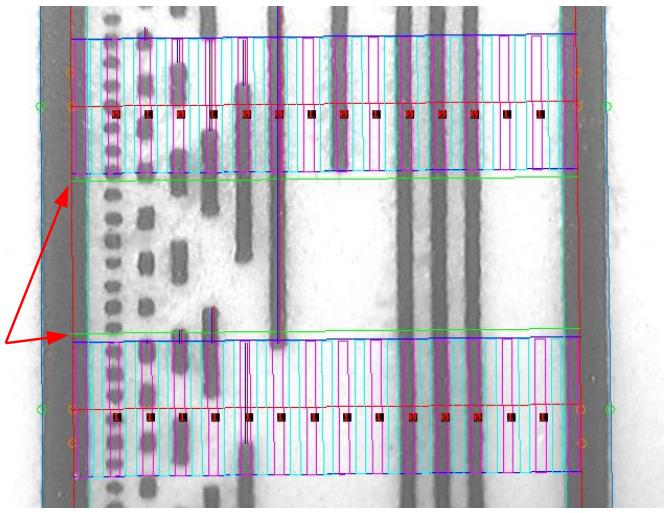
L'ordre de grandeur de l'erreur sur la résolution serait de 1%, si une erreur de 1 pixel est faite tous les 100 pixels. La plus grande conséquence serait uniquement une erreur systématique sur un portion du code.

L'expérience montre que cette erreur est au plus de quelques pixels soit, localement, une incertitude théorique qui correspond à un encodage sur 2^{14} bits au lieu de 2^{16} bits.

5.4) Echantillon d'images du code gray

Comme indiqué dans §5.2., nous devons réaliser un ensemble d'images afin de couvrir l'ensemble du code gray.

Toutefois, il peut exister des discontinuités dans le code, comme c'est le cas au niveau de la jonction entre les différentes extrémités de la bande imprimée.



Afin de pouvoir s'affranchir de ses discontinuités, il faut les placer entre les deux lignes de mesures (ligne 96 et 384). De cette façon le logiciel arrivera à lier les différentes portions de la bande. Les discontinuités doivent être placées entre les lignes vertes de l'image ci-contre. Ces deux lignes ont séparées de 144 pixels sur l'image de la webcam (voir §4.9).

5.5) Algorithme

5.5.1) Assemblage de deux images

On considère deux images. L'algorithme impose que deux images consécutives ne peuvent comporter toutes deux une discontinuité dans le code gray.

Le principe est d'utiliser les deux mesures d'une image pour déterminer une échelle locale entre les graduations du code gray et les distances en pixels. L'expérience montre que l'échelle locale déterminée pour une image est suffisamment proche de l'échelle locale de l'image voisine pour confondre les deux.

Ainsi, il n'est pas utile de mémoriser les images mais uniquement les deux valeurs du code Gray associées aux lignes 96 et 384.

3 cas se présentent :

- Cas général (Aucune image ne présente de discontinuité):

  On note $\boldsymbol{d_e}$ le demi-rapport entre l'échelle en pixels de l'image et l'échelle locale en graduation de la bande.

  On prend comme référence la première image. Ainsi, $d_e = \frac{1}{4}\left(\frac{288}{v_{i,1} - v_{i,0}} + \frac{288}{v_{i,1} - v_{i,0}}\right)$

  avec $v_{i,0}$ et $v_{i,1}$ respectivement première valeur et la seconde valeur des graduations de l'image i donnée par le ligne 96 et 384.

  Remarques:
  On suppose de plus que $v_{i,0} > v_{i,1}$, ainsi $d_e<0$. Cette remarque vaut aussi pour les deux cas suivants.

  L'indice i est tel que $0 \leq i < n$, n étant le nombre de l'image. Si i=n-1, on substitue i+1 par 0.

Ainsi, le nombre de pixels Δy_i entre les deux images est donné par:

$$\Delta y_i = (v_{i+1,0} - v_{i,0} + v_{i+1,1} - v_{1,1})\, d_e$$

- Si la première image présente une discontinuité:

On prend comme référence la seconde image. Ainsi, $d_e = \frac{1}{2} \frac{288}{v_{i+1,1} - v_{i+1,0}}$

et Δy est estimé par: $\Delta y_i = 2(v_{i+1,0} - v_{i,0})\, d_e$

- Si la seconde image présente une discontinuité:

On prend comme référence la première image. Ainsi, $d_e = \frac{1}{2} \frac{288}{v_{i,1} - v_{i,0}}$

et Δy est estimé par: $\Delta y_i = 2(v_{i+1,1} - v_{i,1})\, d_e$

Finalement, le nombre de pixels effectif $\boldsymbol{N_p}$ pour parcourir l'ensemble du disque est donné par:

$$N_p = -\sum_{i=0}^{n-1} \Delta y_i$$

A Chaque image i, on associe la graduation en pixels définie par $G_0=0$, et $G_{i+1}=G_i - \Delta y_i$

5.5.2) Rééchantillonnage

(Etape 1)

On parcourt les n images. On considère l'image i, $0 \le i < n$.

On note ν_0 la fonction qui fait correspondre les valeurs du code gray à la graduation en pixels pour les valeurs associée à la ligne 96 de la webcam et ν_1 l'équivalent pour la ligne 384.

L'objectif de cette première étape est, pour une image i donnée de déterminer les valeurs x_0 et x_{max}, entre les quelles on peut définir les valeur de ν_0 et ν_1 grace aux valeurs $v_{i,1}$ et $v_{i,0}$ de l'image i, puis de tabuler les valeurs du code gray de dixième en dixième en leur associant un numéro la graduation en pixels correspondant.

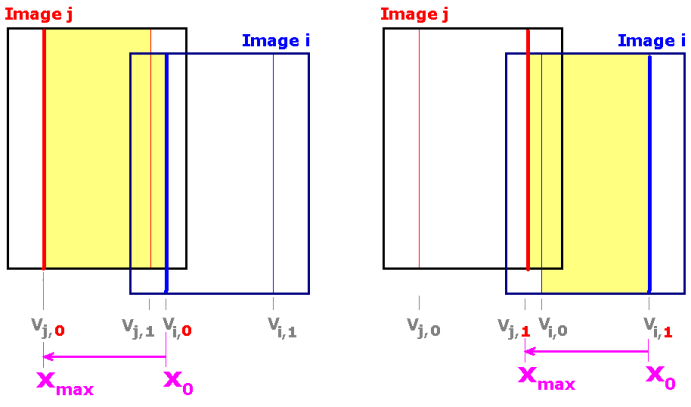
Comme pour le paragraphe précédent, il faut considérer 3 cas en fonction de la présence ou non de discontinuité sur les images.

- Cas général (pas de discontinuité):

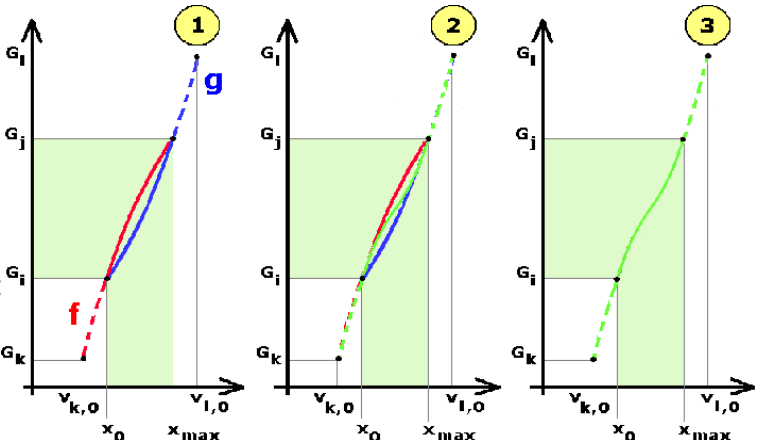
On définit:
k=i-1. Si i=0, on remplace k=i-1 par k=n-1.
j=i+1. Si i=n-1, on remplace j=i+1 par j=0.
l=j+1. Si j=n-1, on remplace l=j+1 par l=0.

Pour ν_0 nous avons: $x_0 = v_{i,0}$ et $x_{max} = v_{j,0}$

Pour ν_1 nous avons: $x_0 = v_{i,1}$ et $x_{max} = v_{j,1}$



Les courbes rouges et bleues de la figure de droite sont respectivement les interpolations f et g sur 3 valeurs réalisée par des fonctions du second degré (voir §5.5.3).



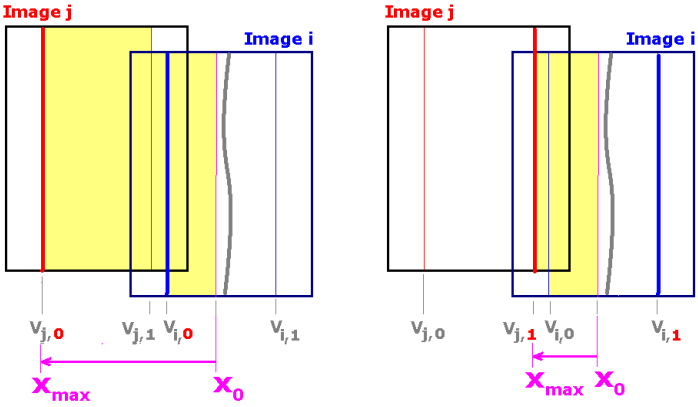
- Si aucune des images i, j, k et l ne comporte de discontinuité, les images de valeur de l'intervalle $[x_0; x_{max}]$ sont déterminée à l'aide de la courbe verte (figure 3 de droite) correspondant à l'utilisation d'une fonction de lissage (voir §5.5.3) appliquée sur les deux fonctions d'interpolation précédente.

Dans ce cas, $\nu_0 = L_{21}\, f + L_{21}\, g$.

- Si l'image k comporte une discontinuité, $\nu_0 = g$.
- Si l'image l comporte une discontinuité, $\nu_0 = f$.

De façon similaire, on détermine ν_1.

- <u>La première image présente une discontinuité:</u>



On définit:
j=i+1. Si i=n-1, on remplace j=i+1 par j=0.
l=j+1. Si j=n-1, on remplace l=j+1 par l=0.
k=l+1. Si l=n-1, on remplace k=l+1 par k=0.

Pour ν_0 nous avons: $x_{max} = v_{j,0}$
Pour ν_1 nous avons: $x_{max} = v_{j,1}$

Compte tenu de la discontinuité de l'image i, nous devons limité la valeur de x_0 dans une zone ne présentant pas de discontinuité. Ainsi, aussi bien pour ν_0 et ν_1 :

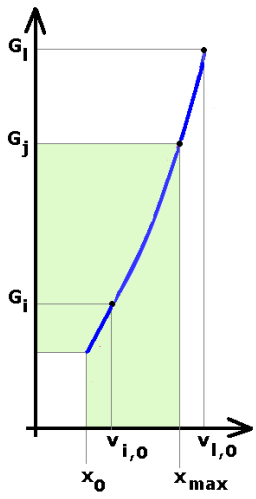
Pour déterminer la valeur de x_0, on tient de la largeur maximum d'une discontinuité G_{max} (et G_{eff}) définie au §4.9. Ainsi,

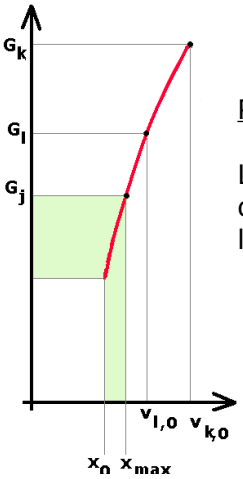
$$x_0 = v_{i,0} - (v_{j,0} - v_{j,1}) \frac{1}{2} \frac{G_{max} - G_{eff}}{288}$$

<u>Pour</u> ν_0 :

La fonction d'interpolation du second degré f (voir §5.5.3) entre les valeurs des images i, j, l sert à extrapoler les valeurs en pixels associées à l'intervalle de valeur $[x_0, v_{i,0}]$.

Ainsi, Sur $[x_0; x_{max}]$ $\nu_0 = f$





Pour ν_1 :

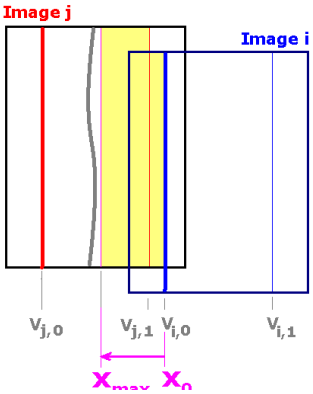
La fonction d'interpolation du second degré g (voir §5.5.3) entre les valeurs des images j, l, k sert à extrapoler les valeurs en pixels associées à l'intervalle de valeur $[x_0, x_{max}]$.

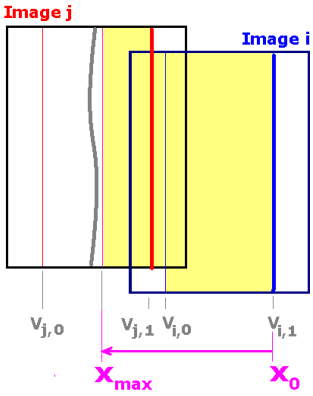
Ainsi, Sur $[x_0; x_{max}]$ $\nu_1 = g$

- La seconde image présente une discontinuité:

On définit:
k=i-1. Si i=0, on remplace k=i-1 par k=n-1.
j=i+1. Si i=n-1, on remplace j=i+1 par j=0.
l=k-1. Si k=0, on remplace l=k-1 par l=n-1.

Pour ν_0 nous avons: $x_0 = v_{i,0}$
Pour ν_1 nous avons: $x_0 = v_{i,1}$





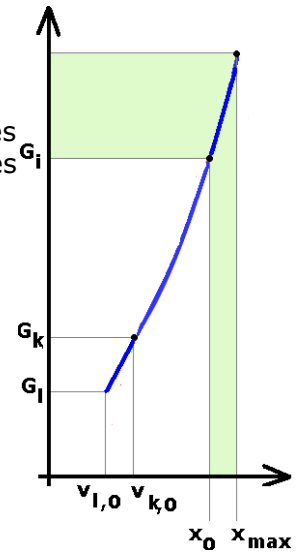
De façon similaire au cas précédent,

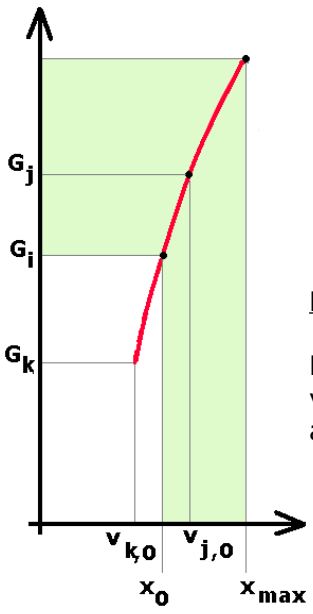
$$x_{max} = v_{j,1} + (v_{i,0} - v_{i,1}) \frac{1}{2} \frac{G_{max} - G_{eff}}{288}$$

Pour ν_0 :

La fonction d'interpolation du second degré f (voir §5.5.3) entre les valeurs des images l, k, i sert à extrapoler les valeurs en pixels associées à l'intervalle de valeur $[x_0, x_{max}]$.

Ainsi, Sur $[x_0; x_{max}]$ $\nu_1 = f$





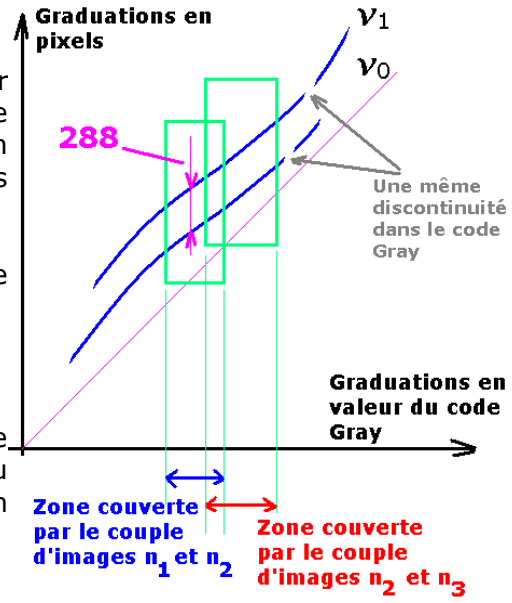
Pour ν_1 :

La fonction d'interpolation du second degré g (voir §5.5.3) entre les valeurs des images k, i, j sert à extrapoler les valeurs en pixels associées à l'intervalle de valeur $[x_0, v_{i,0}]$.

Ainsi, Sur $[x_0; x_{max}]$ $\nu_0 = g$

(Etape 2)

Les courbes associées aux fonctions ν_0 et ν_1 sur l'ensemble des valeurs prisent lors de la lecture du code Gray, sont théoriquement translatées de 288 pixels. En effet, la webcam films la même bande et les deux lignes de lecture sont décalé de 288 pixels.



Dans le cas idéal, en se plaçant en dehors de toute discontinuité, nous avons donc:

Pour toute valeur x du code Gray $\nu_1(x) - \nu_0(x) = 288$

Compte tenu de l'incertitude sur la valeur de la lecture cette valeur de translation après interpolation ou extrapolation peut varier d'une valeur de x (valeur en unité du code Gray)

Pour chaque intervalle de valeur du code Gray couverte par un même couple d'images on calcule la moyenne de la valeur du décalage en pixels entre les 2 courbes à laquelle on soustrait 288.

$$\Delta = \frac{1}{n} \sum_{i=0}^{n-1} \left(\nu_1(x) - \nu_0(x) \right) - 288$$

La valeur Δ indique un erreur de positionnement des deux images couvrant la zone.

Pour les n images couvrantes la bande cyclique du code Gray, nous déterminons ainsi n valeurs de Δ_i . On note Δ_{max} la plus grande valeur de décalage entre 2 images parmi toutes les images.

L'étape suivante va permettre d'optimiser le positionnement de l'ensemble des images entre elles en minimisant la valeur de Δ_{max}.

(Etape 3)

On modifie légèrement la position des images entre elles afin de vérifier si on n'obtient pas un meilleur modèle. Toutefois, l'ensemble des valeurs Δ étant lié compte tenu de la nature cyclique de l'imagerie de la bande, on modifie la position des images entre uniquement d'une fraction aléatoire de la valeur Δ_i que l'on note δ_i.

Si Δ_{max} est inférieur à celui de l'itération précédente, on applique la même fraction à l'ensemble de Δ_i .

Dans ce cas $\delta_i = \lambda \Delta_i$ $0 \leq \lambda \leq 1$ λ déterminer aléatoirement

Sinon on applique une fraction variant aléatoirement pour chaque Δ_i

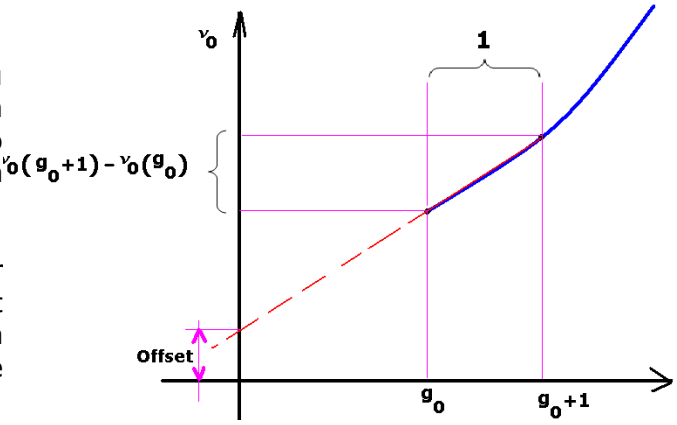
Dans ce cas $\delta_i = \lambda_i \Delta_i$ $0 \leq \lambda_i \leq 1$ λ_i déterminer aléatoirement

On réitère en retournant à l'étape 1 jusqu'à un nombre d'itérations fixé par le fichier de configuration.

(Etape 4)

Nous devons estimer la position du zéro du code Gray afin de l'associer à la graduation zéro en pixels, même dans le cas où le zéro de la bande du code Gray est inexistante en raison de la présence d'une discontinuité.

Nous procédons ainsi: On note α la valeur opposée à l'offset correspondant, avant correction à la graduation en pixels de la valeur zéro du code Gray (voir graphique de droite)



On note g_0 la valeur la plus petite du code Gray balayée lors de l'étape 1. Afin d'optimiser le code en utilisant des tableaux, g_0 est dans la pratique la partie entière de 10 fois la valeur du code Gray. g_0 est donc une valeur entière. On défini l'offset de la façon suite:

$$\alpha = g_0 \frac{\nu_0(g_0+1) - \nu_0(x)}{1} - \nu_0(g_0)$$

(Etape 5)

On parcourt l'ensemble des valeurs balayées lors des étapes précédente afin d'associer à chaque x (valeur du code gray à la graduation en pixels correspondant et en tenant compte de la correction de l'offset.

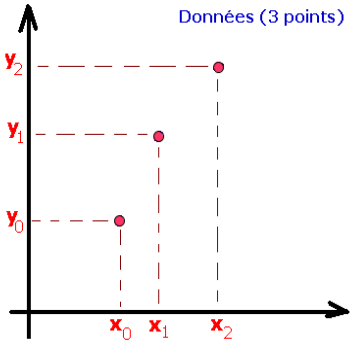
Hors discontinuité, on substitue donc $\nu_0(x)$ par $\nu_0(x)+\alpha$ et $\nu_1(x)$ par $\nu_1(x)+\alpha$

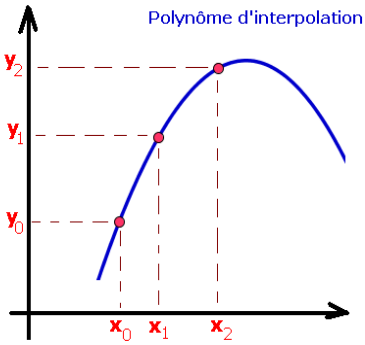
(Etape 6)

L'étalonnage est sauvegardé en prenant soin préalablement d'associer a chaque graduation en code gray sa valeur en radian pour la lecture suivante la première ligne de lecture (ligne 96) ou la seconde ligne de lecture (ligne 384). De même on précise la graduation du code gray ayant la même signification angulaire lue suivant la première ou la seconde ligne.

5.5.3) Fonction d'interpolation et fonction de lissage

- La fonction f servant à l'interpolation et l'extrapolation dans l'étape 1 est un polynôme d'interpolation de Lagrange du second degré.





Ainsi,

$$f(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} y_0 + \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} y_1 + \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} y_2$$

- La fonction de lisage L utilisée dans l'étape 1 vérifie les conditions suivantes:

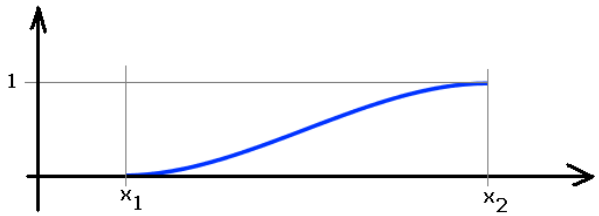
$$\begin{cases} L_{12}(x_1)=0 \\ L_{12}(x_2)=1 \\ L'_{12}(x_1)=0 \\ L'_{12}(x_2)=0 \end{cases}$$

En choisissant L comme un polynôme du troisième degré nous obtenons le système:

$$\begin{cases} L'_{12}(x)=k(x-x_1)(x-x_2) & \text{où k est une constante} \\ L_{12}(x)= P(x)(x-x_1)^2 & \text{où } P(x)=ax+b \quad \text{avec a et b deux constantes} \\ L_{12}(x_2)=1 \end{cases}$$

Ainsi, on obtient l'expression de L,

$$L_{12}(x) = \frac{-(2x + x_1 - 3x_2)(x - x_1)^2}{(x_2 - x_1)^3}$$



VI) Effets de la dilatation

6.1) Présentation

La dilatation des supports ne peut être évitée. Le disque supportant la bande imprimée est a priori composé d'une matière différente de celle de la bande. Ceci crée une dilatation différentielle entre les deux matériaux.

Si on ne prend pas garde à l'effet de dilatation, l'étalonnage du code gray, devra être fait a chaque changement conséquent de température.

Dans cette partie nous allons voir comment limiter l'effet de la dilatation.

A noter que la dilatation modifie les longueurs mais pas les angles. Ainsi, les portions angulaires repérées par les graduations du code gray ne change pas dans la mesure que la dilatation se produit de façon homogène sur l'ensemble du code.

6.2) Contrainte sur la fixation de la bande imprimée

Pour compenser la dilatation différentielle entre le disque et la bande imprimée, il ne faut pas coller la bande sur le disque. La localisation des jonctions créées par la colle entre les deux matériaux n'est pas métrisable. On ne peut garantir l'homogénéité du collage et les effets élastiques localisés du collage.

Ainsi, la bande sera montée tendue sur le disque. Le code gray sera découpé en 4 secteurs afin d'assurer une tension homogène sur chaque secteur.

De plus, si la bande imprimée se dilate plus rapidement que le disque la supportant, la bande risque de friser. A l'inverse, si le disque se dilate plus rapidement que la bande, celle dernière peut potentiellement se rompre. Toutefois, on peut considérer un support relativement élastique pour l'impression du code gray.

L'élasticité d'un matériau est proportionnelle à la force de traction appliquée dans la limite élastique.

On choisi d'imprimer le code gray sur du papier calque indéchirable en polyester. Afin d'alléger le support, le disque est réalisé en aluminium. Le code sera imprimé sur une feuille A3 en polyester afin de s'attribuer de la réserve sur la bande afin d'avoir des prises pour tendre celle-ci.

Nous avons réalisé un essai en traction afin de mesurer le module d'élasticité de Young du film polyester utilisé. Nous avons obtenu une valeur ***E*** de 2500 Mpa. On considère un coefficient de dilatation α_p de 100 $\mu m.m^{-1}.°C^{-1}$ pour le polyester et α_a de 24 $\mu m.m^{-1}.°C^{-1}$ soit une valeur différentielle de 76 $\mu m.m^{-1}.°C^{-1}$.

Afin d'éviter que la bande ne soit détendue pour la plus grande température T_{max}, celle-ci sera montée suffisamment tendue à la température de montage T_0.

L'allongement différentielle entre la bande et le support ΔL_d est donné par :

$$\frac{\Delta L_d}{L} = \alpha\,(T_{max}-T_0) \qquad \text{avec} \quad \alpha = \alpha_p - \alpha_a$$
$$\text{et} \quad L \text{ est la longueur de la bande}$$

L'allongement ΔL_e dû à l'élasticité de la bande est donnée par :

$$\frac{\Delta L_e}{L} = \frac{F}{SE} \qquad \text{avec} \quad F \text{ la tension de la bande}$$
$$\text{et} \quad S \text{ la section de la bande}$$

Ainsi, si $\Delta L_d = \Delta L_e$ nous avons:

$$\alpha (T_{max}-T_0) = \frac{F}{SE}$$

d'où $$T_{max} = T_0 + \frac{F}{\alpha SE}$$

Exemple:

	Dilatation (µm/m/°C)	Module de Young (Mpa)
Polyester	100	2500
Aluminium	24	69000
Différence	**76**	

Température de montage	20	°C
Tension de montage	16	N
Largeur du ruban	15	mm
Épaisseur du ruban	80	µm
Section du ruban	*1,20E-006*	*m²*
T_max_Théorique	*90,18*	*°C*

Ainsi, pour un température de montage de 20°C, la bande doit être tendue à 16N pour qu'elle puisse resté tendu jusqu'à 90°C.

Remarque: On retrouve ce tableau de calcul dans le fichier « calculs_encodeur_polyester.xls »

6.3) Effet sur la lecture du code

Le paragraphe précédent garanti la tension du support du code. Toutefois cette tension n'est que longitudinale. Ainsi, du point de vue de la webcam, la dilatation suivant la circonférence du disque n'est pas la même que suivant la direction perpendiculaire.

La lecture du code repose sur la connaissance du rapport e_x et e_y défini au §4.2.

e_x va subir la dilatation du au disque d'aluminium alors que e_y va suivre la dilatation du au film en polyester.

Or $$\frac{e_x}{e_y}\,\frac{e_y + \Delta e_y}{e_x + \Delta e_x} = \frac{1+ \alpha_p \Delta T}{1+ \alpha_a \Delta T}$$

Soit avec $\Delta T = 30°$ $\alpha_p = 100\ \mu m.m^{-1}.°C^{-1}$ et $\alpha_a = 24\ \mu m.m^{-1}.°C^{-1}$

On obtient $$\frac{e_x}{e_y}\,\frac{e_y + \Delta e_y}{e_x + \Delta e_x} = 1{,}0022$$ soit 0,22% dévolution du rapport

La lecture d'une valeur du code gray est effectuée en observant une zone de 128 pixels, e_x ayant un ordre de grandeur de 4 pixels, on peut observer 32 fois l'échelle de e_x sur cette zone .

Or $(1{,}0022-1) \times 128 \approx 0{,}3$ pixel

Une erreur de lecture de 0,3 pixel est de l'ordre de 7% d'une graduation soit un ordre de grandeur en-dessous de l'échelle e_x. Cet effet dû à la dilatation ne sera donc pas problématique.

VII) Effet de la vitesse de rotation

La vitesse de rotation du code gray devant l'objectif de la webcam, compte tenu de la vitesse d'acquisition d'une image, est équivalant à l'application d'un filtre faisant disparaître le basse fréquence. Ainsi, on perd la pertinence de la valeur lue par webcam sur les bits de poids faibles, mais l'algorithme présenté précédemment arrivera à lire le bit de poids forts, donnant donc une valeur moins précise mais cohérente de la position angulaire.

Ceci n'est pas gênant car pour un déplacement rapide nous pouvons nous contenter d'une précision sur le positionnement moindre.

A mesure que l'on se rapproche de la position désirée la vitesse de déplacement diminue et donc la précision de la lecture s'améliore.

Ainsi, l'effet d'une vitesse de rotation importante n'est pas problématique.

Finalement, pour une application astronomique, la vitesse de rotation de l'axe d'ascension droite est suffisamment faible pour ne pas être perceptible pendant l'acquisition d'une image, il n'y aura donc pas de perte de résolution à cette vitesse.

VIII) Hardware

Dans le cadre d'une utilisation de plusieurs webcams à la fois, il faut préciser un point.

Pour une application en astronomie, nous avons besoin de mesurer la position angulaire de deux axes: l'axe horaire et l'axe des déclinaisons. Il nous faut donc deux webcams.

Usuellement, une webcam occupe 70% des capacités d'un contrôleur USB. Il faut donc un contrôleur USB par webcam, quel que soit le système d'exploitation (Windows, Linux etc...)

Un contrôleur USB peut gérer plusieurs ports USB. Typiquement, pour une carte USB il y a un contrôleur. Sur certaine carte mère, il peut y avoir plusieurs contrôleurs USB mais ce n'est pas toujours le cas.

Ainsi, pour brancher une seconde webcam, il se peut qu'il faille acheter un carte USB supplémentaire bien qu'il y ait des ports USB encore disponibles sur la machine.